Mission Critical Systems for Security Management

William H. Sawyer, Ph.D.

G. S. Butz, Inc.

Exton, Pennsylvania 19341

NDIA Conference

Williamsburg, Virginia

June 16, 1998

Mission Critical Systems for Security Management

Dr. William H. Sawyer, Ph.D.

Abstract

Virtually all modern security management systems use distributed processing so that the system continues to function when the CPU is down. For medium and high security applications this is not sufficient. The CPU is the critical interface between human and machine. Loss of the CPU blinds the operator to critical events and key system information. For this reason the CPU is as mission critical as the alarm panels and the intelligent door controllers. The most familiar operating systems for PC's (the preferred CPU (server) for most security management systems today) are not mission critical. Windows 95, Windows NT, OS2 and various PC versions of UNIX all can create significant data addressing problems if they fail during operation. While it is possible to recover from most of these problems, the recovery time is significant and requires a level of expertise not typically expected of a security system operator. This paper outlines the nature of these problems, suggests steps which can be taken by security managers and system designers to prevent loss of critical event notification during a significant recovery time, and outlines solutions which will greatly reduce the probability of such a problem occurring. This paper is vendor neutral. Various systems manufacturers and their attempts to address this problem are not discussed.

Mission Critical Systems for Security Management

William H. Sawyer, Ph.D.
G. S. Butz, Inc.
Exton, Pennsylvania 19341

I Introduction

Mission Critical and Security Management are terms which go together in the minds of most people responsible for the security of their facilities. Each of us has a picture in our own minds of what such a system looks like or should look like. Yet these pictures usually conflict with reality, and in some cases the picture and reality are mutually exclusive. In this paper I would like to take you through some of the issues which create these dilemmas and suggest some ways around them. The importance of these dilemmas to you and their solutions, each of which involves some type of constraint, usually money, you will have to decide.

Experience has taught us as a manufacturer that whether or not it is in the specification, every customer wants their system to run all the time no matter what the circumstance. There are no acceptable excuses. People will accept "the system is down . . . " from there is departments. They will accept it from the travel agent. They will even accept it from their bank. They cannot and will not accept it from their security management system.

At the same time each customer wants all the features of a modern computer system such as a Graphical User Interface (GUI) sometimes simply referred to as Windows. They: you, want it to run on PC's just like everything else, in fact you want it to run on your brand of PC since your company or agency has a purchasing agreement with someone and you can get it for much less than the security dealer will sell it to you.

In some cases you even want it to run on your own LAN/WAN or at least one which uses the same operating system (OS) so that your people do not need to learn two systems. (To purchase a service contract from the security vendor is to give them a license to steal.) Yet you want complete security from internal or external hackers.

These are just some of the constraints faced by us and our competitors as major security systems manufacturers today. They make UL, NFPA, and BOCA requirements seem trivial by comparison. No one, not even us, offers you the answer to all of these problems simultaneously in an inexpensive package which will fit every budget. The intent of this paper is to try to highlight some of the pitfalls and traps which are associated with the above requirements and illustrate various ways around them. Using a broad brush, we will rank the solutions we present according to their effectiveness, difficulty to implement and maintain, and initial cost. Your specific requirements may lead to a very different result.

II Security Management System Functions

Historically the two major functions of security management systems have been alarm monitoring and access control. Some years ago closed circuit television (CCTV) was added so that today most systems integrate all of these features in some way. More recently personnel management functions such as Video Badging, Time and Attendance, and Visitor Registration have been added to these systems. See Figure 1. Today many manufactures are adding low voltage lighting control, HVAC, and system maintenance management capabilities to their systems. In the future it is likely that at least certain forms of communication will also be included. The impetus to do this has come from many directions:

- The recognition that each of these areas has a critical security component,
- There can be a resulting operating cost reduction,
- There can be an increase in overall system reliability if these new functions are implemented using the design architecture developed by the security industry,
- The increasing requirement by customers for security systems to have an open architecture which has led to the implementation of some features of the BackNet protocol and LON Works.
- The increased requirement for mission critical secure communications with system analysis which will record, track, and analyze attempted security breaches.
- In order to simplify training, operation, and maintenance, there must be a similar look and feel to each system.

III The need for mission criticality

It is intuitively obvious that the systems above should all be hardened against failure. They should be able to remain operational under the most difficult of circumstances. Lightning, critical component failure, CPU failure, and operator error, to name a few should not be able to compromise or bring the system or any one of its major components down.

The security industry recognized this early in its career. Battery backup and un-interruptible power supplies (UPS) have always been available. Distributed processing was introduced in the early 1970s with the advent of microprocessors which made possible intelligent local controllers. Redundant communication schemes and two ended loop communications were introduced at nearly the same time. Software which drove the intelligent controllers in these systems was written in the assembly code native to the microprocessor which was the heart of the system. It was a forgone conclusion that this code had to be written in such a way that it did not crash. In addition, should the microprocessor itself experience a glitch, a “watchdog” circuit could immediately reboot the system without loss of data and the entire operation could continue. Files could be updated in real time and the system's data and logs could be backed up without the need to interrupt the system's operation.

These were the days of the Z80, CPM, and later DOS. As good as they were, there were serious limitations. Systems were slow, the sizes of the systems were limited by the constraints of DOS, the cost of memory, and in general they were effectively single user systems. Such features as multiple users and multitasking were at best in the realm of minicomputers which pushed the cost of such systems out of reach of most potential users. Perhaps more important was the fact that every system was different. In many cases even an upgrade from the same manufacturer was incompatible with the previous release of the system.

The development of new operating systems for the PC led to many advances in the security industry. Notable among these were the advent of multiuser, multitasking operating systems and 32 bit internal and external bus architecture. But nothing was to change the landscape as much as the advent of Microsoft Windows in its various versions from Windows 3.1 to Windows 95 and Windows NT. For Microsoft the intent was to provide the PC user with an intuitive, easy-to-use man/machine interface similar to the one so successfully marketed by Apple Computer. Success was swift. The public liked the product and Windows 3.x quickly replaced DOS and its competitors as the PC operating system of choice, even though it ran on top of DOS. Windows 3.x had many serious limitations however which kept it from entering the serious security systems market. It offered virtually no file security, it was not a true multitasking operating system, and it was prone to periodic and unpredictable failures.

The rest of the computer market also became quickly aware of Windows 3.x limitations. Microsoft already had an answer in preparation: Windows NT for network and large system users, and Windows 95 for everyone else. Much effort was put into the development of Windows NT. Among the central design features were the requirements to meet the stringent demands of the Federal Governments C2 and C3 security requirements, and the need for portability which would allow the system to run on many different computers and architectures. Much of this development took place in the early 90s when 16 bit internal architecture and 8 bit and 16 bit external architectures were in general use.

In order to accommodate these and other requirements, compromises inevitably had to be made. Among these was the loss of mission criticality. The most obvious manifestation of this is the requirement to go through a shut down procedure when you want to turn off a computer using either Windows 95 or Windows NT. Where DOS was an extremely robust operating system rarely suffering a fatal failure, Windows in all its versions has been subject to fatal failures many of these as a result of its memory utilization scheme. If a DOS program did fail, it was possible to reboot the system immediately. If any data was lost, it was only that which was being written to the disk at the time of the interruption. As you all have painfully experienced at one point or another, such is not the case with Windows. When it is restarted, at best the system must go through the procedure of reexamining the hard disk files while it rebuilds the file allocation table or its equivalent.

Let me assure you. This paper is not a plea for the "good old days" of DOS. We are far better off today than we were even two years ago. Rather it is my intent to point out some of the design limitations of current systems and offer you the security system user suggestions which will allow you to overcome them in your system design. As examples, let us examine two of these compromises and their impact on the mission criticality of the modern security management system.

1 Memory allocation schemes:

a. Fixed Partitioning (Figure 2a): In this case a specific region of memory is assigned to each program which should be sufficient for all of its requirements. In some variations where there are large memory requirements, paging may be used where portions of the program or its data file are kept on disk and only that information which is currently required is kept in memory.

Fixed partitioning can be done manually when the system is setup, either by the program when it is started, or by the operating system itself. In any case once a block of memory is allocated to a particular program, only that program and its data can reside there. In early DOS and DOS like systems the partition size was limited to 640K or less depending upon the requirements of the OS (in multitasking environments each program's memory was limited). Later systems allow access to up to several GB. The disadvantage of this scheme is its inefficient use of memory. If a program infrequently requires large amounts of memory, the maximum amount of memory it requires is none the less always allocated to it. 32 bit multitasking multi-user operating systems utilizing this type of memory allocation are UNIX, LINIX, DRDOS, and CCIDOS among others.

b. Dynamic Partitioning(Figure 2b): Here the operating system keeps track of the amount of memory each program needs and the addresses where it is located. As the programs run and their need for memory changes, the operating system reallocates memory to each of them. Since there is rarely ever enough memory, particularly in larger systems, paging schemes are stilled employed. This is a very egalitarian method which results in highly efficient use of memory and in the case of large programs frequently a significant increase in speed.

Unfortunately the technique is not entirely fool proof. Among other problems, for example, a key location which is still needed can be written over by another program or another part of the same program, the operating system believing that the information in that location was no longer required. This error can result in the famous words "Your program has committed a fatal error . . . " Windows operating systems and the newer versions of most other operating systems utilize this technique of memory management.

2 Disk File Allocation (Figure 3): When data is stored on a disk the address or addresses of its location or locations are maintained in a table. There are various techniques for storing this data: the File Allocation Table (FAT) was used by DOS-BASED systems and some implementations in Windows. Windows NT uses the NTFS. OS/2 uses HDFS. In any case, in early DOS and DOS like systems the FAT was stored on the disk at all times and updated when information was written to the disk. With the advent of multitasking systems and the need for larger, faster network operating systems it became the custom to maintain the FAT or its equivalent in memory reducing the number of disk writes by half and greatly increasing the overall speed of the system. This is why it is necessary to go through the shut down procedure on your Windows 95 or Windows NT system.

The problem here is that if a system crashes, it cannot simply be restarted immediately and take up where it left off. The "FAT" must be rebuilt. Most modern operating systems have means of storing the data locations within the data itself so that the table can be relatively easily rebuilt. Unfortunately during this process, the system is down. No critical information can be reported to the operator during this period. If there are extensive files on multiple disks which need to be searched, the process of rebuilding the "FAT" can be a very time consuming process.

To be sure there have been major advances in operating systems which in other ways have greatly improved their reliability. An excellent example is the realtime kernels used in Unix and NT which prevent higher level programs from accessing the processor functions directly thereby greatly reducing the opportunity for system crashes caused by application programs.

Nonetheless as systems grow larger and the market's demand for more user friendly systems increases, so do the complexity of our systems and the opportunity for failure.

IV Mission Critical System Design

To some extent mission criticality is in the eye of the user. To the information systems manager it is the need to protect the files under all circumstances. To the security manager the data files are indeed critical, but even more important is the immediate reporting of critical events. Security information is time critical. You can't take any action until you know an incident has occurred. If the file server or CPU is down, you don't know what is happening. In order to determine the best system design for you begin with an analysis of your needs and a system risk assessment.

1 What is the primary purpose of your system?
  - a Will it be monitored all the time?
  - b Are there life safety issues involved?

2 How big is your system now and how large might it get?
  - a. How many user stations will you need?
  - b Will you be on your own network?

3 What type of alarms will you monitor with your system?

4 How many doors will you control? What kind of control do you need?

5 What other functions will the system serve?

6 How will you use the data stored by your system? i.e.: How important is it?

In even the simplest system timely information is usually the most important issue. Here the criteria are clear.

1 The system should use distributed processing so that if one component fails the entire system doesn't fail.

2 The CPU should be dedicated to the security function so that other operations cannot compromise the systems reporting and logging capability.

3 Use a reliable operating system which is not prone to unpredictable failures. Windows 3.x is not a good candidate. Windows 95 can also be a poor choice.

Larger systems offer more complex problems. Clearly the most important criterion is the timely reporting of critical events followed closely by reliable logging and secure databases. If the system is to have multiple workstations as is the case with most large systems, a client server LAN/WAN is the most likely approach. Many design criteria need to be considered for such a system. Only some of the most important can be discussed here.

1 Hardware and System considerations (Figure 4)

  a. Whenever possible use a separate server and LAN for your security system. It is obviously critical that you control when the server is operational and who has access to it. Although both Windows NT and NetWare have excellent security protection which can be augmented by third party hardware, if it is not your hardware, and it is used by others at the same time, they must necessarily have some say in its operation and maintenance. In other words if enough people want it turned off, it will be turned off.

b There should be redundant means of receiving critical information. This can take several forms:

i Redundant Servers, one operating as a hot backup for the other. Unfortunately Windows NT does not support a hot backup. Manual intervention is required to promote a backup server to be primary server. There is third party software available which claims to provide this capability. Both NetWare 4.1 and the RISK version of UNIX do support a hot backup. For this reason alone, NetWare is the preferred NOS for PC based security management systems.

ii In some system designs as a backup procedure in case of server failure, the Local Area Controllers (LAC) can be automatically accessed by a work station forming a peer to peer network. In other designs, a work station can talk to each of the LAC simultaneously logging information and reporting critical events to the operator.

c Redundant Subsystems: In many cases it is simply too expensive to use redundant servers. In these cases subsystems which might fail are often set up in a redundant configuration. One of the more common subsystems for which redundancy is provided is the hard drive(s). Frequently a redundant array of inexpensive disk drives (RAID) is used. In this case data is written to a pair of drives simultaneously. If one of the drives fails, it can be replaced while the system is operating and then data from the remaining drive is transferred to the new drive so that redundancy is restored.

Redundant routers and bridges may also be required, particularly if the entire information pathway passes through one unit.

d Redundant communication paths are nearly as important as redundant servers. A redundant communication path should have no points in common except the LAC and the Server. In both cases the communications ports should obviously be different. Examples of secondary communications paths include the following:

i A secondary copper or fiber line.
ii An RF link
iii A cellular telephone

In the latter two cases encryption should be considered.

2. Operating System Considerations

Some of the best mission critical operating systems for PC s are known only to the most ardent software experts and hobbyists. For better or for worse, the current market wisdom says that Microsoft is the only company which is going to still be around in five years, so if you don't want your investment to be obsolete, base it on a Microsoft operating system. Yet Microsoft does not make a mission critical multitasking operating system. It doesn't even make an operating system which will support a hot backup. To further complicate the matter, the marketplace demands a standard graphical user interface and that standard is Windows.

Industry rumors have it that Windows NT is slowly moving toward a UNIX like architecture which will provide hot backup capability. The NT kernel is already mission critical and the system is preemptive. But what is the solution now.

The most effective PC architecture is a NetWare 4.1 redundant server configuration using Windows NT Workstation or Windows 95 as the workstation operating system. The Local Area Controllers should run their own mission critical OS such as a POSIX compliant UNIX derivative for PCS, a multitasking DOS type system such as DRDOS, CCIDOS or if they are not using multitasking, standard MSDOS. See Figure 5.

The LAC must maintain a copy of the databases which pertain to their functions. They should support redundant communication paths and be able to be re-booted by a watch dog circuit. Communications between the servers and the LAC must be secure. There should be no way for a workstation to communicate to a LAC if it is not specifically designed and set up to do so.

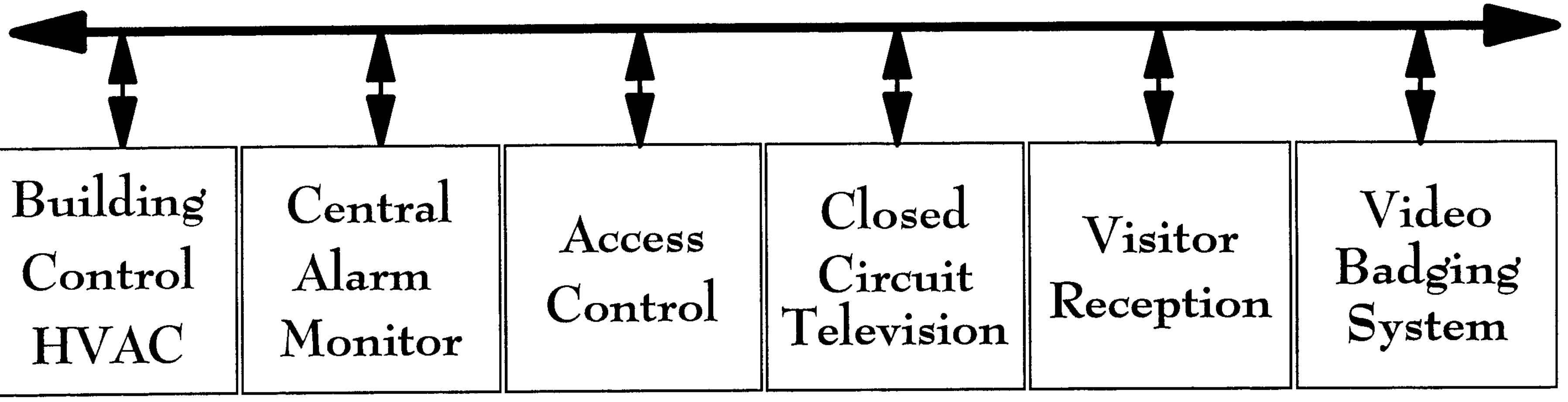
VI, Summary

Early security management systems could be designed to be mission critical. The programming was done in a language close to the processor, so the software could be designed to be highly fault tolerant. But each manufacturer's system was different with no common architecture, and no expansion capability to include other related functions which were not explicitly incorporated by the manufacturer. Improvements in PC operating systems led to a "standard" graphical user interface and a more common architecture. However, as a result of the changes in operating system design which led to these features, it was not possible for the system manufacturers to maintain open architecture and mission criticality simultaneously. Program engineers were forced to write code which did not directly interact with the processor. Now they were dependent upon the behavior of the operating system which lay between their code and the processor itself.

Mission critical systems today require a hybrid approach to system design. If the server system is to utilize a hot backup, it must either use a version of Novell NetWare, or combination Windows NT 4 and third party software. The workstations will utilize either Windows 95, or Windows NT Workstation. The Local Area Controllers will maintain their own databases as well as transmit them to the server. They will support redundant communication links to a server and utilize a mission critical operating system which in general does not support a GUI.

The future will most likely see the evolution of mission critical operating systems for servers. Windows NT is already moving in this direction as is Novell. It is too early to determine whether or not Novell will withstand the Microsoft onslaught.

Modern Security Management System

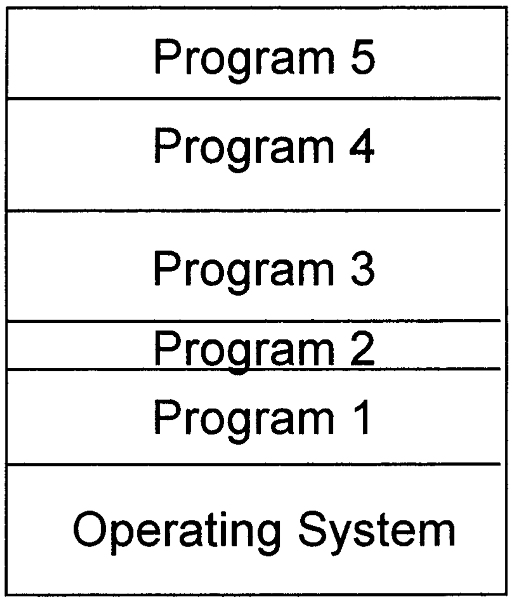


Features	Benefits
Open Architecture	Works with multiple vendors systems The end user can select the best System for each function
PC Based	Easily Serviced, Easy to use, Inexpensive Extensive Software Available
Unified Database	Maximum report generation ability

Figure 1

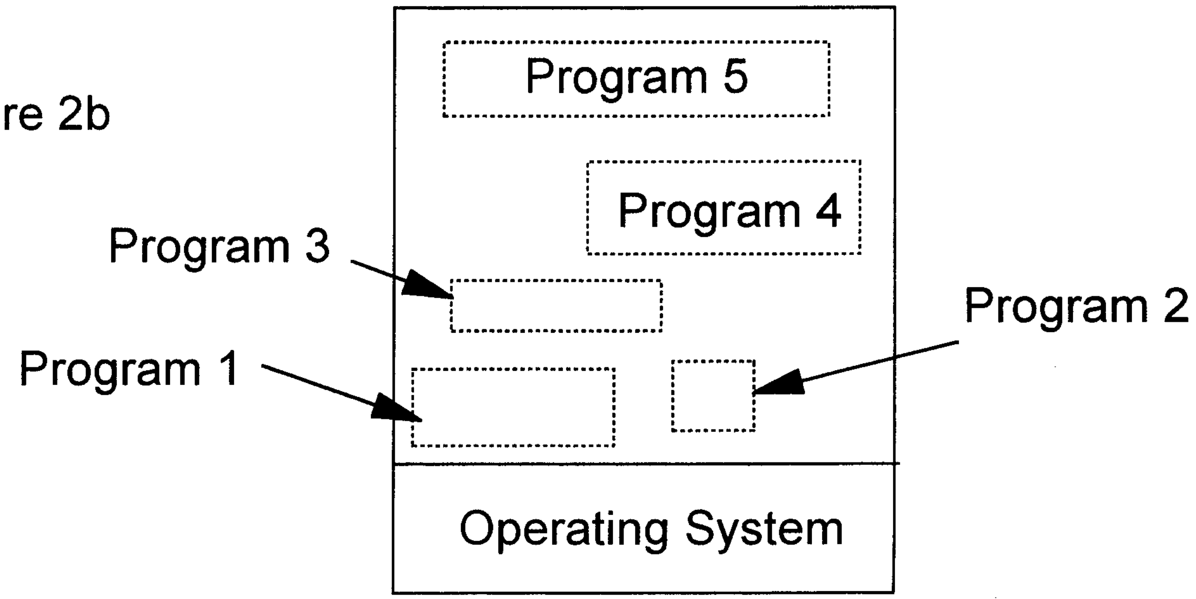
Fixed Partition Memory Map

Figure 2a

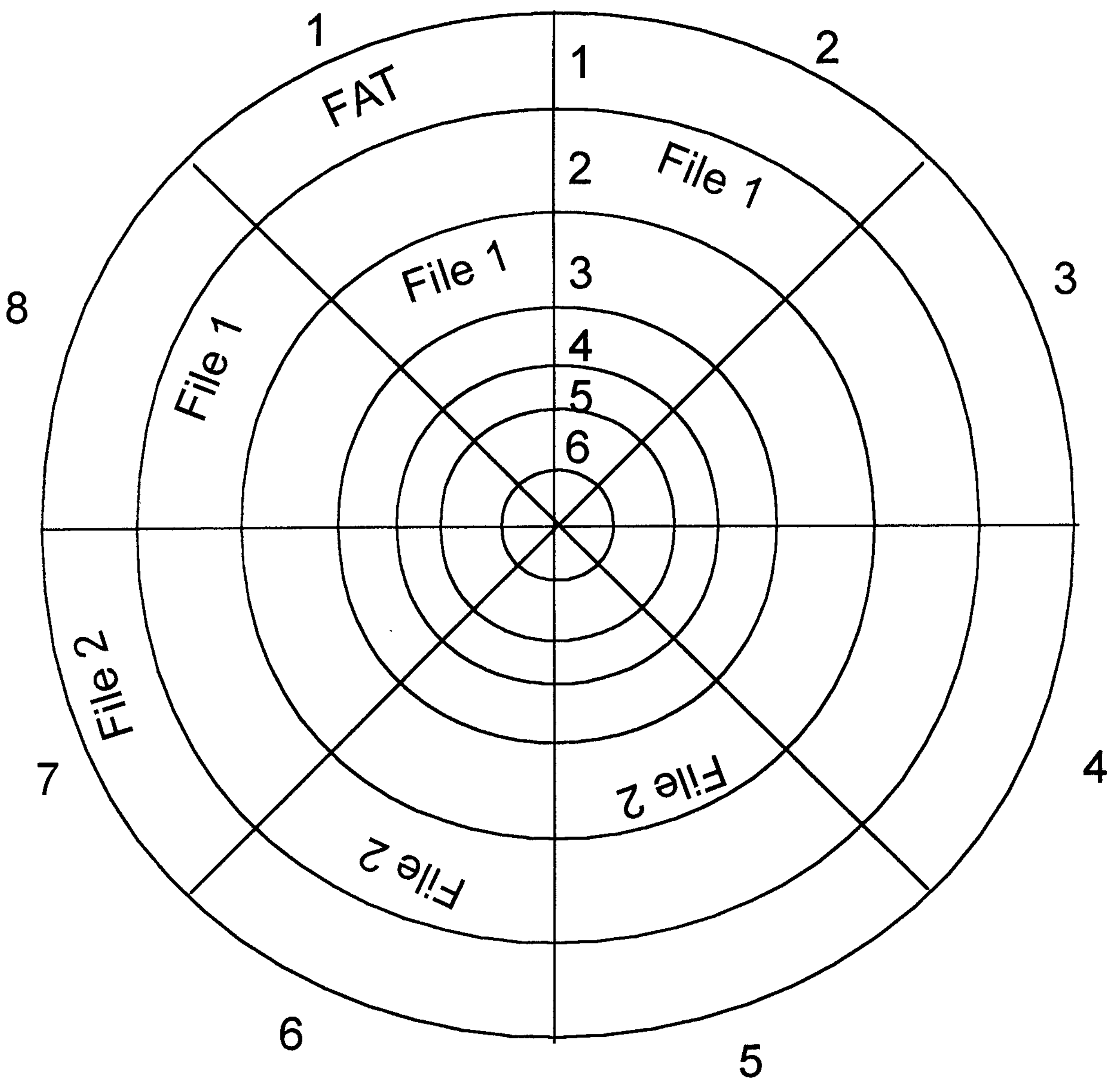


Dynamic Partition Memory Map

Figure 2b



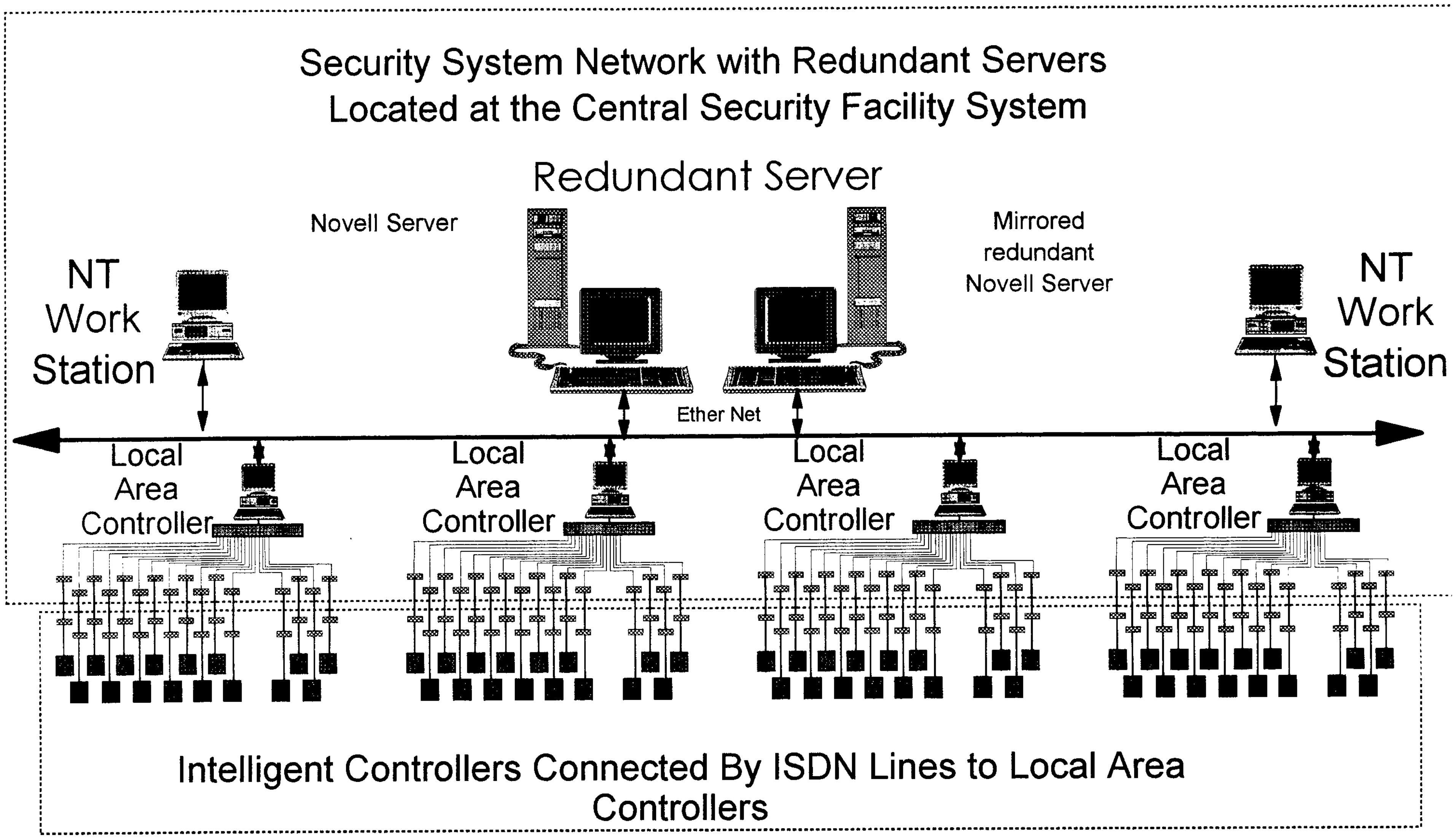
Physical Disk Data Distribution



File Allocation Table

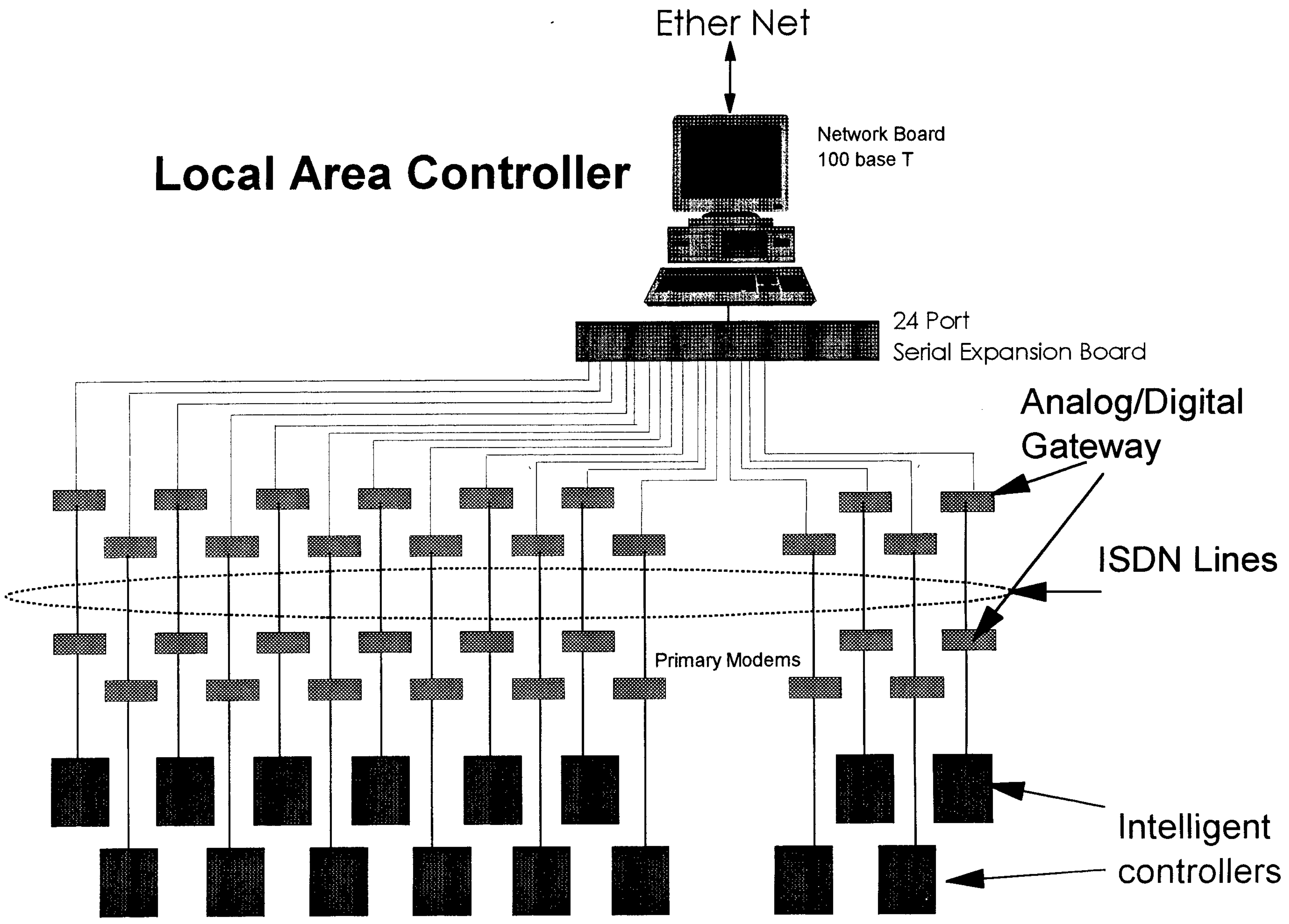
File	Sector	Cylinder
1	8	2
	1	3
	2	2
2	5	3
	6	2
	7	1

Figure 3


Security System Network with Redundant Servers
Located at the Central Security Facility System
Redundant Server
Novell Server
Mirrored redundant Novell Server
NT Work Station
NT Work Station
Ether Net
Local Area Controller
Local Area Controller
Local Area Controller
Local Area Controller
Intelligent Controllers Connected By ISDN Lines to Local Area Controllers

Mission Critical Security Management System

Figure 4



Local Area Controller
Figure 5